

Colour- Assembler

Copyright (c) 1983 by Trommeschläger Computer GmbH
Geschrieben von Kalle Braun

Programm und Dokumentation sind urheberrechtlich geschützt

Ein Programm für den Colour-Genie EG 2000 mit 16K-RAM, 32K sind
empfehlenswert.

1. Ladehinweise	Seite 2
2. Einleitung	Seite 3
3. Der Assembler	Seite 4
a) Eingabe eines Maschinenprogramms	Seite 4
b) Starten des Übersetzungsvorgangs	Seite 7
c) Optionen zur Beeinflussung der Übersetzung	Seite 8
d) Fehlermeldungen und ihre Ursachen	Seite 10
e) Besondere Assemblerbefehle	Seite 11
4. Anwendungsbeispiel	Seite 13

Ladehinweise:

Der Colour-Assembler ist in Maschinensprache geschrieben und kann mit dem SYSTEM-Befehl eingelesen werden:
Schalten Sie Ihr Colour Genie ein und drücken Sie (RETURN).
Nachdem das Colour-Basic READY erschienen ist, geben Sie SYSTEM ein und drücken (RETURN). Der Rechner gibt '*?' aus, geben Sie nun COLASM ein und bereiten Sie Ihren Kassettenrekorder auf Wiedergabe vor. Dann drücken Sie (RETURN). Der Rechner wird das Programm nun von der Kassette in den Speicher lesen. Während dem Ladevorgang erscheinen in der oberen rechten Ecke des Bildschirms zwei Sternchen, wobei das rechte Sternchen langsam blinkt, während das linke ruhig stehenbleibt. Falls sich dieses linke Sternchen in ein 'C' verwandelt, liegt ein Ladefehler vor, und Sie sollten den Ladevorgang mit den beiden RST-Tasten abbrechen und die Lautstärke- und Tonkopfeinstellung Ihres Rekorders überprüfen. Falls es Ihnen auch bei guter Einstellung des Rekorders nicht gelingen sollte, das Programm fehlerfrei zu laden, bitten wir Sie, uns die Originalkassette in Originalverpackung zurückzusenden. Sie erhalten dann eine neue Kopie.
Wenn der Ladevorgang fehlerfrei verläuft, erscheint nach seinem Ende wieder '*?'. Nun geben Sie '/' ein und drücken (RETURN).
Der Rechner löscht den Bildschirm und meldet sich mit:
COLOUR-ASSEMBLER
(C) 1983 BY TCS

READY

>

Nun ist der Colour-Assembler einsatzbereit.

Einleitung:

Dieses Handbuch setzt voraus, das Sie mit Maschinensprache einigermaßen vertraut sind. Sollte dies nicht der Fall sein, so empfehlen wir Ihnen einschlägige Literatur.

----- WICHTIG -----
Die Firma Trommeschläger kann nicht für Schäden verantwortlich gemacht werden, die durch die Anwendung dieses Programms an anderen Programmen entstehen können.

Der Colour-Assembler ist ein Programm, das Ihnen die Eingabe von Maschinensprache sehr stark erleichtert, da es Ihnen die Mühe abnimmt, die "Mnemonics" in echte Maschinensprache (die Opcodes) umzuwandeln.

ACHTUNG: Der Colour-Assembler belegt den Speicher Ihres Colour-Genies von 5800H bis ca. 7100H; das bedeutet, daß Sie bei der 16 K-Version noch etwa 3 3/4 kByte für Ihr Quellprogramm, die Cross-Reference Tabelle und das übersetzte Programm frei haben, also ein maximal 768 Bytes langes Maschinenspracheprogramm schreiben können. Dies ist nicht allzu viel, reicht aber, um sich etwas in der Anwendung von Maschinensprache zu üben. Trotzdem empfehlen wir Ihnen, sich die Speicherkarte, die Ihren Speicher auf 32 K erweitert, zu kaufen, um ein sinnvolles Arbeiten mit dem Colour-Assembler sicherzustellen.

Eingabe eines Maschinenprogramms:

Die Eingabe von Maschinensprache-Quelltexten erfolgt mit dem normalen, Ihnen bekannten Colour-Basic Editor, d.h. sie brauchen sich nicht an einen anderen Editor zu gewöhnen. Aus diesem Grund entfällt auch ein langes Kapitel über den Editor in diesem Handbuch, da Sie alles darüber im Handbuch 'COLOUR-BASIC leicht gelernt', das Ihrem Colour Genie beilag, nachlesen können.

Hier jedoch noch ein Kapitel über den Aufbau des Quelltextes: Der normale Aufbau eines Maschinensprachekommandos ist:

(SYMBOL) BEFEHL (OPERANDEN) (KOMMENTAR)

1. Symbole

Ein Symbol ist ein Name für eine absolute Adresse. Symbole müssen nicht in jeder Zeile stehen, es ist jedoch sinnvoll, sie immer dann zu verwenden, wenn ein bestimmtes Programmsegment von einer anderen Stelle angesprungen werden soll. Ein Symbol ist eine Zeichenkette mit maximal 6 Zeichen, wobei das erste Zeichen ein Großbuchstabe sein muß, während die anderen Großbuchstaben oder Ziffern sein müssen. Ein Symbol darf folgende Zeichenfolgen nicht enthalten:

\$: Dieses Zeichen bedeutet die augenblickliche Adresse. A; B, C, D, E, H, L, I, R, IX, IY, SP, PC, AF, BC, DE und HL

Diese Zeichenfolgen stellen die Register des Z80-Mikroprozessors dar.

C, Z, NC, NZ, PO, PE, P und M

Diese Zeichenfolgen stellen die 8 Abfragemöglichkeiten des Statusregisters der Z80 dar

(Siehe auch das Anwendungsbeispiel auf Seite 13)

Ein Symbol muß direkt an der ersten Stelle der Zeile beginnen. Wenn an dieser Stelle ein Leerzeichen steht, wird angenommen, daß diese Zeile kein Symbol enthält.

2. Befehle

Die vom Colour-Assembler verarbeiteten Z80-Befehle entsprechen genau den im 'Z-80-Assembly Language Programming Manual, 3.0 D.S., REL. 2.1, FEB 1977' aufgeführten. Wir empfehlen auch hier einschlägige Literatur.

3. Argumente

Argumente sind ein oder zwei Werte, die durch ein Komma voneinander getrennt sind. Manche Befehle benötigen auch überhaupt keine Argumente.

Beispiel:

LD HL,4400H

XOR A

LD (HL),20H

Ein Wert in Klammern () bedeutet indirekte Adressierung, wenn sie mit Registern benutzt werden, sonst 'Inhalt von'. Eine Konstante ist eine Zahl, die mit folgenden Buchstaben enden darf:

H - Die Konstante ist hexadezimal

O - Die Konstante ist oktal

D - Die Konstante ist dezimal

Eine Zahl, die mit keinem dieser Buchstaben endet, wird als dezimal angenommen.

Konstanten müssen mit einer Ziffer beginnen: FFH ist

verboten. 0FFH wird richtig erkannt. Beispiele:

3C00H
 16384D
 16384
 7750
 (Siehe auch Kap. 'Operatoren')

4. Kommentare

Alle Kommentare müssen mit einem Semikolon beginnen. Ein Kommentar wird bei der Übersetzung ignoriert, er dient nur dazu, Gedankengänge beim Schreiben des Quelltextes zu vermerken. Wenn ein Semikolon als erstes Zeichen in einer Zeile steht, wird die ganze Zeile als Kommentarzeile betrachtet.

OPERATOREN

Der Wert eines Arguments kann sich auch aus einem Ausdruck berechnen. Ein solcher Ausdruck kann die '+', '-', '&'- und '>'-Operatoren enthalten. Ein Ausdruck wird streng von rechts nach links berechnet, Klammern sind nicht erlaubt.

1. Addition

Das Pluszeichen addiert zwei Konstanten und/oder Symbole. Wenn ein Plus alleinsteht, wird nur der Wert wiedergegeben:

```
001E      CON30  EQU  30
0010      CON16  EQU  10H
0003      CON3   EQU  3
4400      VIDEO  EQU  4400H
4403      A1     EQU  VIDEO+CON3
002E      A2     EQU  CON30+CON16
4400      A3     EQU  +VIDEO
```

2. Subtraktion

Das Minuszeichen subtrahiert zwei Konstanten und/oder Symbole voneinander. Ein einzelnes Minus ergibt das 2er-Komplement:

```
43FD      A1     EQU  VIDEO-CON3
000E      A2     EQU  CON30-CON16
BC00      A3     EQU  -VIDEO
```

3. Logisches Und

Das Undzeichen verknüpft zwei Konstanten und/oder Symbole logisch miteinander, d.h. im Ergebnis ist ein bit nur dann gesetzt, wenn das entsprechende bit in beiden Argumenten gesetzt war:

```
4400      A1     EQU  VIDEO&OFFH
0000      A2     EQU  0&15
2222      A3     EQU  6666H&0AAAAH
```

4. Schiebebefehl

Das '>'-Zeichen verschiebt eine Zahl nach rechts oder links. Die Form ist:

ZAHL > WERT

Wenn WERT eine positive Zahl ist, wird ZAHL nach links, sonst nach rechts, geschoben:

```
4000      A1     EQU  VIDEO > 4
0440      A2     EQU  VIDEO > -4
```

Zusammenfassung:

```
0440      A2      EQU  VIDEO+15 > -4
44F0      A3      EQU  VIDEO > -4+15 > 4
Hierbei ist die Reihenfolge zu beachten:
VIDEO > -4 = 0440H
0440H+15 = 044FH
044FH > 4 = 44FOH
```

Starten des Übersetzungsvorgangs:

Nachdem Sie Ihren Quelltext mit dem Colour-Basic Editor eingegeben haben, können Sie den Übersetzungsvorgang starten. Dies geschieht mit dem Befehl 'NAME' (dieser Befehl liegt auch auf Taste F3), dem noch einige Optionen folgen können (siehe auch Seite 8). Der Bildschirm wird gelöscht, und folgender Text erscheint auf dem Bildschirm: COLOUR-ASSEMBLER
(C) 1983 BY TCS

DURCHGANG NR. 1 (SYMBOLTABELLE)

Nach einiger Zeit (diese Dauer ist abhängig von der Länge des Quelltextes) erscheint darunter:
DURCHGANG NR. 2 (ASSEMBLIERUNG)

Danach erscheint (wiederum nach einer von der Länge des Quelltextes abhängigen Zeitdauer):

LAENGE DES QUELLTEXTES : xxxxx
LAENGE DER SYMBOLTABELLE : xxxxx
LAENGE DES OBJECTCODES : xxxxx

(RETURN) FUER SYMBOLTABELLE

Wenn Sie nun die (RETURN)-Taste drücken, wird die alphabetisch sortierte Symboltabelle ausgegeben. Wenn eine Seite voll ist, wird 'BITTE DRUECKEN SIE EINE TASTE' ausgegeben und gewartet. Wenn Sie eine beliebige Taste drücken, wird die Ausgabe fortgesetzt. Dies geht solange weiter, bis die alle Symbole ausgegeben wurden, dann erscheint wieder:
'BITTE DRUECKEN SIE EINE TASTE'

Wenn Sie nun eine Taste drücken, wird zum Colour-Basic READY zurückgesprungen.

(siehe auch Seite 9).

Optionen zur Beeinflussung der Übersetzung:

Zur Beeinflussung des Übersetzungsvorgangs stehen drei Optionen zur Verfügung, die folgendermaßen angewählt werden:

NAME/option/option....

1. OA

Durch Eingabe dieser Option wird nach der Übersetzung und der Ausgabe der Symboltabelle der Name abgefragt, unter diesem das übersetzte Programm auf Kassette aufgezeichnet werden soll. Dieser Name darf maximal 6 Zeichen lang sein, das erste Zeichen muß ein Buchstabe sein, alle anderen dürfen Buchstaben oder Ziffern sein. Nach dieser Eingabe erscheint *** KASSETTE FERTIG? (RETURN) ***. Bereiten Sie Ihren Kassettenrekorder auf Aufnahme vor und drücken Sie die (RETURN)-Taste. Nun wird das übersetzte Programm aufgezeichnet. Das so beschriebene Band können Sie mit dem SYSTEM-Befehl des Colour-Basic wieder einlesen und starten.

ACHTUNG: OA kann und darf nicht zusammen mit der Option TM verwandt werden!!!!!!

2. NS

Durch Eingabe dieser Option wird die Ausgabe der Symboltabelle unterdrückt.

3. TM

Durch Eingabe dieser Option werden bei der Übersetzung alle ORG-Befehle ignoriert und das gesamte Programm so übersetzt, daß es an der Stelle, an die der Colour-Assembler das übersetzte Programm ablegt, sofort lauffähig ist. Hierbei muß allerdings beachtet werden, daß das Programm einfach so übersetzt wird, als ob am Anfang ein entsprechendes ORG stände, d.h. nur die Symbole werden entsprechend angepaßt. Wenn Sie also ein Programm, das 250 Bytes lang ist, durch ein ORG 5800H ab der Adresse 5800 hex ablegen wollen, und einen Speicher benötigen, in dem Sie eine Eingabe o.ä. ablegen wollen, so können Sie natürlich sagen:

Das Programm beginnt bei 5800H und ist 250 Bytes lang, also lege ich den Speicher ab 5900H an. Wenn Sie ein solches Programm mit der Option TM übersetzen lassen, wird diese Adresse natürlich nicht verändert und es kann Ihnen passieren, daß Ihr Programm bei der Ausführung den Colour-Assembler zerstört. Aus diesem Grunde ist es ratsam, sämtliche Adressen, die in Ihrem Programm benutzt werden, durch Symbole zu kennzeichnen. Dasselbe gilt natürlich für den END-Befehl (siehe Seite 11). Ein Programm, das bei Adresse 5800H beginnt und mit einem END 5800H aufhört, wird auch nach der Option TM als bei 5800H liegend gestartet. Wenn Sie die Option TM gewählt haben, so wird nach der Übersetzung und der Ausgabe der Symboltabelle folgender Text ausgegeben:

'(RETURN) FUER PROGRAMMAUSFUEHRUNG'

Drücken Sie die (RETURN)-Taste, um Ihr Programm zu starten. Hierbei ist zu beachten, daß der Colour-Assembler automatisch die Adresse 0066H (den Start der Basic-Hauptschleife) auf dem Stack ablegt, sodaß Ihr Programm auch mit dem Befehl RET abschließen kann.

ACHTUNG: Die Option TM kann und darf nicht mit der Option OA zusammen verwandt werden!!!!!!

Außerdem gilt, daß Sie bei jeder Abfrage durch Druck auf die (BREAK)-Taste zum Colour-Basic READY zurückkehren können.

Fehlermeldungen und ihre Ursachen:

Der Colour-Assembler erkennt 9 verschiedene Fehler:

1. ZU WENIG SPEICHERPLATZ IN ZEILE xxxxx
Der vorhandene Speicherplatz reicht nicht mehr aus, um das gesamte Programm zu übersetzen. Falls dieser Fehler schon während des ersten Durchgangs auftritt, ist es unmöglich, das Programm durch Kürzungen noch passend zu machen, da schon die Symboltabelle über die Speichergrenzen hinausging. Wenn dieser Fehler in Durchgang Nr. 2 auftritt, sollten Sie versuchen, das Programm durch Benutzung von Unterprogrammen, weniger Symbolen u.ä. zu kürzen. Hierbei sollten Sie allerdings bedenken, das auch die Option OA (Aufzeichnung des übersetzten Programms) zusätzlichen Platz benötigt.
2. NICHT DEFINIERTES SYMBOL IN ZEILE xxxxx
In der genannten Zeile wurde ein Symbol benutzt, daß nicht definiert wurde.
3. DOPPELT DEFINIERTES SYMBOL IN ZEILE xxxxx
In der genannten Zeile wurde ein Symbol zum zweitenmal definiert.
4. FEHLERHAFTE OPTIONEN-WAHL
Es wurde eine Option gewählt, die nicht existiert (nur OA, NS, und TM existieren) oder OA und TM wurden zusammen gewählt.
5. KEIN END-BEFEHL IM PROGRAMMTEXT
Der Quelltext wurde nicht mit einem END-Befehl abgeschlossen.
6. INKORREKTER AUSDRUCK IN ZEILE xxxxx
Eine Konstante stimmt nicht mit dem ihr folgenden Buchstaben überein:
 - Ein Hexzahl wie 1A19 wird nicht von einem H gefolgt.
 - Eine Ziffer einer Zahl darf nicht größer oder gleich der Basis des Zahlensystems sein (80 , 1GH, OA)
7. NICHT DEFINIERTER BEFEHL IN ZEILE xxxxx
Ein Befehl, der nicht zu den Z80-Befehlen gehört, wurde gefunden (LOAD HL,5800H u.ä.).
8. NICHT ERLAUBTE ADRESSIERUNGSART IN ZEILE xxxxx
Ein Befehl hat zuwenig oder nicht passende Argumente:
LD HL,BC
LD D,(BC)
LD BC
9. ZU WEITER RELATIVER SPRUNG IN ZEILE xxxxx
Ein relativer Sprung darf höchstens 128 Bytes zurück- oder 127 Bytes vorwärtsspringen.

Besondere Assemblerbefehle:

Das \$-Zeichen wird im Colour-Assembler dazu benutzt, den augenblicklichen Wert des 'Assemblerpointers' wiederzugeben. Dieser Assemblerpointer enthält die Speicheradresse, an die das nächste Byte theoretisch geschrieben würde. Theoretisch deshalb, weil das Programm ja nur bei TM wirklich an die Stelle geschrieben wird, an der es auch laufen soll.

Beispiel:

```
10      ORG  5800H
20      LD   HL,$
würde als LD   HL,5800H übersetzt,
10      ORG  5800H
20      LD   HL,$
30      LD   HL,$
würde als LD   HL,5800H
          LD   HL,5803H übersetzt.
```

Der Colour-Assembler kennt 8 besondere Befehle, die nicht direkt übersetzt werden, sondern zur Steuerung der Übersetzung dienen:

1. ORG nn
Alle folgenden Befehle werden so übersetzt, daß das Programm im Adressenbereich ab nn lauffähig ist, d.h. der Assemblerpointer wird gleich nn gesetzt (wird von TM ignoriert).
2. EQU nn
Weist dem vorstehenden Symbol einen Wert zu. Darf nur einmal pro Symbol auftauchen.
3. DEFL nn
Weist dem vorstehenden Symbol einen Wert zu. Kann mit demselben Symbol und verschiedenen Werten mehrmals auftauchen (Umdefinieren eines Symbols).
4. END nn
Ein END-Befehl muß am Ende des Quelltextes stehen. Der Wert nn (ein Symbol oder ein Ausdruck oder eine Zahl) gibt die Einsprungsadresse in das Programm an.
5. DEFB n
Das Byte n wird in der Speicherzelle, auf die der Assemblerpointer zeigt, abgelegt.
6. DEFB 'c'
Der ASCII-Wert des Zeichens c wird in der Speicherzelle, auf die der Assemblerpointer zeigt, abgelegt.
7. DEFW nn
Das LSB des Zwei-Byte Wortes nn wird in der Speicherzelle, auf die der Assemblerpointer zeigt, abgelegt, das MSB in der nächsten Speicherzelle.
8. DEFS nn
nn Bytes werden ab der Speicherzelle, auf die der Assemblerpointer zeigt, reserviert, d.h. der Wert des Assemblerpointers wird um nn erhöht.

9. DEFM 'zeichenkette'

Die Zeichenkette innerhalb der beiden Hochkommas wird im Speicher ab der Speicherzelle, auf die der Assemblepointer zeigt, abgelegt.

Anwendungsbeispiel:

Das hier abgedruckte Programm dient zur Umrechnung von Dezimalzahlen in Hexzahlen. Sie werden aufgefordert, eine Dezimalzahl (max. 5 Ziffern) einzugeben, und das hexadezimale Äquivalent dieser Zahl wird ausgegeben. Wenn Sie bei der Eingabe (BREAK) drücken, wird zum Colour-Basic READY zurückgesprungen:

```

10      ORG 5800H
20 START CALL 01C9H      ; BILDSCHIRM LOESCHEN
30 START1 LD HL,INMSG
40      CALL PRTMSG
50      LD HL,NUMBU
60      LD B,05H
70      CALL 0040H      ; INLINE-ROM-ROUTINE
80      JP C,0066H
90      EX DE,HL
100     LD HL,0000H
110 LOOP1 LD A,(DE)
120     CP 30H
130     JR C,ENDE
140     CP 3AH
150     JR NC,ENDE
160     LD C,L      ; HL = 10 * HL
170     LD B,H      ;
180     ADD HL,HL    ;
190     ADD HL,HL    ;
200     ADD HL,BC    ;
210     ADD HL,HL    ;
220     SUB 30H
230     LD C,A
240     LD B,00H
250     ADD HL,BC
260     INC DE
270     JR LOOP1
280 ENDE LD A,H
290     CALL SBYTE
300     LD A,L
310     CALL SYBTE
320     LD A,0DH
330     CALL PRTBYT
340     JP START1
350 SBYTE PUSH AF
360     RRCA
370     RRCA
380     RRCA
390     RRCA
400     CALL NIBBLE
410     POP AF
420 NIBBLE AND 0FH
430     ADD A,90H
440     DAA
450     ADC A,40H
460     DAA
470 PRTBYT PUSH DE
480     CALL 0033H
490     POP DE
500     RET
510 PRTMSG LD A,(HL)
520     OR A

```

```
530          RET    Z
540          CALL  PRTBYT.
550          INC   HL
560          JR    $-7
570 INMSG    DEFM  'DEZIMALZAHL?'
580          DEFB  00H
590 NUMBU    DEFS  5
600          END    START
```

Wenn Sie dieses kleine Programm eingegeben haben, geben Sie ein:

NAME/TM/NS

Der Bildschirm wird gelöscht und es erscheint:

COLOUR-ASSEMBLER

(C) 1983 BY TCS

DURCHGANG NR. 1 (SYMBOLTABELLE)

DURCHGANG NR. 2 (ASSEMBLIERUNG)

LAENGE DES QUELLTEXTES : 1232

LAENGE DER SYMBOLTABELLE : 84

LAENGE DES OBJECTCODES : 115

(RETURN) FUER PROGRAMMAUSFUEHRUNG

Drücken Sie (RETURN) und der Bildschirm wird gelöscht. Der Rechner fragt nun eine Dezimalzahl ab. Geben Sie ein:

DEZIMALZAHL? 65535

Der Rechner antwortet mit:

FFFF

DEZIMALZAHL?

Diese Spiel können Sie solange fortsetzen, wie Sie wollen. Wenn Sie auf die Frage DEZIMALZAHL? die (BREAK)-Taste drücken, erscheint wieder:

READY

>

Da das vorangegangene Programmbeispiel doch recht komplex ist, sei hier noch ein zweites, einfacheres Programmbeispiel aufgelistet. Das Programm schreibt den Bildschirm weiss und wartet anschließend, daß die (RETURN)-Taste gedrückt wird. Dann wird der Bildschirm wieder gelöscht und in's Basic zurückgesprungen. Zum Weisschreiben des Bildschirms wird Grafikzeichen CHR\$(202) benutzt. Die Kommentare (angefügt mit einem ";") müssen natürlich nicht mit eingegeben werden.

```
10      ORG      7F00H
20 ANF   LD      A,202      ; Akkumulator mit 202 laden
30      LD      (4400H),A   ; und in 4400H schreiben
40      LD      HL,4400H    ; Register für ; HL: von
50      LD      DE,4401H    ; LDIR-Befehl ; DE: nach
60      LD      BC,960      ; laden. ; BC: Länge
70      LDIR     ; Blocktransfer (weisschr.)
80 TAST  LD      A,(0F840H) ; (RETURN) abfragen:
90      CP      1          ; gedrückt, wenn Bit 0 von
100     JP      NZ,TAST     ; Adresse F840H gesetzt.
110     CALL    01C9H       ; ROM-Routine für CLS
120     JP      0066H       ; Rücksprung in's Basic
130     END      ANF
```

Nach Eingabe des Programms, geben Sie, wie auf der vorigen Seite erklärt, NAME/TM/NS ein und das Programm wird assembliert.